HIGH SPEED A/D DSP INTERFACE FOR CARRIER DOPPLER TRACKING

IN-32
046 726

Timothy Baggett

A technical report in partial fulfillment of the requirements for the Degree Master of Science in Electrical Engineering

NMSU-ECE-98-010

Advisor: Prof. Phillip L. De Leon
New Mexico State University
Klipsch School of Electrical and Computer Engineering
Box 30001, Dept. 3-0
Las Cruces, New Mexico 88003

HIGH SPEED A/D DSP INTERFACE FOR CARRIER DOPPLER TRACKING

Timothy Baggett

NOVEMBER 1998 NMSU-ECE-98-010

Abstract

As on-board satellite systems continue to increase in ability to perform self diagnostic checks, it will become more important for satellites to initiate ground communications contact. Currently, the NASA Space Network requires users to pre-arrange times for satellite communications links through the Tracking and Data Relay Satellite (TDRS). One of the challenges in implementing an on-demand access protocol into the Space Network, is the fact that a low Earth orbiting (LEO) satellite's communications will be subject to a doppler shift which is outside the capability of the NASA ground station to lock onto. In a prearranged system, the satellite's doppler is known a priori, and the ground station is able to lock onto the satellite's signal.

This paper describes the development of a high speed analog to digital interface into a Digital Signal Processor (DSP). This system will be used for identifying the doppler shift of a LEO satellite through the Space Network, and aiding the ground station equipment in locking onto the signal. Although this interface is specific to one application, it can be used as a basis for interfacing other devices with a DSP.

Acknowledgments

This work was supported in part through NASA Grant NAG 5-1491 and donations from Motorola, SPS, to New Mexico State University. Special thanks are in order to Phil Brewer for generating the schematic drawings in Appendix C.

Table of Contents

List of Tables

List of Figures

List of Source Listings

Glossary of Acronyms

A/D	Analog to Digital Converter
ADS	Application Development System
BPSK	Binary Phase Shift Keying
CNR	Carrier-to-Noise Ratio
CODEC	Coder/DECoder, a combined A/D and D/A
D/A	Digital to Analog Converter
DAMA	Demand Assignment Multiple Access
DMA	Direct Memory Access
DSP	Digital Signal Processor, or Digital Signal Processing
EMI	External Memory Interface
EVM	Evaluation Module
FFT	Fast Fourier Transform
GPIO	General Purpose Input or Output
I/O	Input and/or Output
IRQ	Interrupt Request as in IRQA, IRQB, etc.
ISR	Interrupt Service Routine
LEO	Low Earth Orbit
NASA	National Aeronautics and Space Administration
NFET	N channel Field Effect Transistor
PC	Personal Computer, generically Microsoft Windows based
PCB	Printed Circuit Board
PEROM	Programmable Erasable Read Only Memory

PFET	P channel Field Effect Transistor
RAM	Random Access Memory
SR	Status Register
SRAM	Static Random Access Memory
SSI	Synchronous Serial Interface
TDRS	Tracking and Data Relay Satellite
VBR	Vector Base Register

1. Overview

1.1. Introduction

The goal of this project is to interface an analog to digital (A/D) converter to a digital signal processor (DSP) for the purpose of implementing a real-time DSP-based recovery of a carrier with an unknown doppler shift. This DSP-based carrier doppler extraction is to be utilized in the NASA Space Network. In a more generic sense, this project can be used as a guide for developing other real-time high-speed DSP applications.

The NASA Space Network consists of two geostationary Tracking and Data Relay Satellites (TDRS) that relay data from low Earth orbiting (LEO) satellites. The user LEO satellite communicates to the NASA ground station through one of the geostationary TDRS. In the past, NASA has provided communication services to the LEO satellite operators through TDRS by pre-scheduling. A spacecraft is only able to communicate with the ground station during these pre-scheduled times. At times, a LEO satellite may in fact require on demand contact with the ground station. One such event may be the on-board detection of a potential spacecraft fault. In such a case, the ability for the NASA to allow on-demand access to the Space Network would be valuable to the LEO satellite operators.

As the LEO satellite orbits around the Earth, its transmissions to the TDRS geostationary satellite will appear to have a doppler shift. In pre-scheduled transmissions, the location of the LEO with respect to the TDRS will be known. The NASA Space Network ground station has the capability of locking on to the LEO signal, if the frequency of the doppler shifted LEO signal is known to within 3 KHz. Once locked, the ground station will continue tracking the LEO signal, as the doppler shift changes during its orbital pass under TDRS.

In a demand assignment multiple access (DAMA) system, the LEO may request access to the Space Network without a pre-scheduled time. In this event, the ground station may not know in advance the position of the LEO with respect to the TDRS, and therefore the expected doppler shift. As discussed in the technical report "Doppler Extraction For a Demand Assignment Multiple Access Service For NASA's Space Network" [Sanchez], the maximum expected DAMA signal doppler shift is +-64 KHz. Since this is outside of the 3 KHz ability for the ground station to acquire and lock onto the DAMA signal, without knowledge of the doppler shift, the ground station would be unable to establish a communications link. As described in the technical report by Monica Sanchez and Dr. Stephen Horan, a DSP can be used to determine the doppler shift on the DAMA signal within the ground station receiver passband. Once the doppler shift is known, the ground station can lock onto the DAMA signal.

2. Problem Definition and Approach

2.1. DAMA Project Requirements

The proposed DAMA signal is a direct sequence spread spectrum Binary Phase Shift Keyed (BPSK) signal, spread at 100K chips/sec. Therefore the null-to-null bandwidth is approximately 200 KHz. Since this signal, transmitted by a LEO satellite, may experience a doppler shift of +-64 KHz, the DAMA signal mainlobe may lie within a 328 KHz frequency range. This frequency band will be sampled by an analog to digital (A/D) converter and the doppler estimated by the DSP.

Dr. Phillip De Leon and NASA White Sands Ground Terminal staff measured the DAMA signal carrier to noise ratio (CNR) at 45 dB. Therefore, the A/D used in the DAMA carrier doppler extraction should provide adequate resolution so as not to introduce noise into the system.

2.2. DSP Selection

2.2.1. Initial DSP requirements calculation

In order to obtain the specified sample rates and effectively process the signal, the processing speed of the DSP must be considered. The Motorola DSP56303 was identified as the appropriate DSP to be used for this application. The DSP56303 is a member of Motorola's 56300 family DSP. The DSP56300, or 'Onyx', based DSP family is capable of executing one instruction per clock cycle. Operating with an 80 MHz clock, the DSP56303 is capable of 80 MIPS (million instructions per second).

At high sample rates, the amount of processing the DSP can do between each sample becomes critical. At 80 MIPS, the DSP will be able to execute approximately 100 instructions during each sample period at 800 Ksps. This was determined to provide an adequate average amount of processing for each sample for this application. The DSP would buffer a burst of samples during a short time period, then halt the sampling of the signal to process the buffered samples.

The DSP56303EVM was chosen for this application for it's relatively high features and low cost at $199. The DSP56303EVM comes with an on board high quality stereo audio CODEC capable of 8 or 16 bit samples at a rate up to 44.1, 32Kx24 fast external SRAM, and most DSP signals are brought to connectors on the board for easy integration with user developed hardware. Although the onboard CODEC is satisfactory for audio processing applications, it is unsuitable for applications that require a higher sampling rate, outside of the normal audio frequency spectrum. Therefore, an alternate A/D will be required to process and estimate the doppler on the DAMA signal.

2.2.1.1. SSI vs. External Memory Interface

The typical audio CODEC interfaces with the DSP via a full duplex synchronous serial interface, or SSI. The data samples are transferred between the CODEC and DSP in a serial fashion (clocked one bit at a time). At audio rates, a serial transfer of the data is sufficient. However, at high sample rates, the SSI simply may not be fast enough.

The DSP56303 has a maximum SSI clock rate of 12.5 Mbps. With an 800 Ksps sample rate, the SSI would be able to transmit only 15 bits of date per sample period. For this reason, most high-speed A/D or D/A devices utilize a parallel interface. These parallel A/D or D/A converters transfer the entire sample through a parallel data bus.

2.3. A/D Selection

As derived in a technical report by Monica Sanchez and Dr. Stephen Horan (1996), the required processing bandwidth of the DAMA signal for this project is 328 KHz. The Nyquist sampling theorem specifies the minimum sampling rate, Ω_S, must be at least twice the bandwidth of the signal, Ω_N, to be sampled to insure the signal is uniquely sampled. If the Nyquist sampling theorem is not met, then the sampled signal will be distorted as portions of the original frequency spectrum will 'spill over' into other areas of the spectrum. To satisfy the Nyquist Sampling theorem for the DAMA project, we therefore must choose an A/D that will sample at a rate of at least twice 328 Ksps, or 656 Ksps, to prevent distortion of the signal.

$$\Omega_S > 2\Omega_N \qquad (1)$$

When an analog signal is sampled, its quantized amplitude value will typically be different from the actual analog amplitude at the instant the sample was made. An A/D converter that makes samples B bits wide has 2^B levels of quantization. During the sampling process, an A/D will round down the actual analog amplitude to the nearest quantization level. The difference between the actual amplitude and the sampled amplitude adds quantization noise to the system. By increasing the number of bits per sample an A/D can make, the number of quantization levels is increased. This makes the quantization levels smaller, and reduces the quantization noise injected by the system. Oppenheim and Schaffer show that quantization noise is uniform. They also show that for each extra bit added per sample, the signal to quantization noise is increased by approximately 6 dB [Oppenheim and Schafer].

The DAMA signal was measured by White Sands Ground Station Staff and Dr. Phillip De Léon of the New Mexico State University Center for Space Telemetry and Telecommunications. They found that the signal had a minimum 45 dB carrier to noise ratio (CNR). Therefore, to prevent the addition of significant quantization noise in the system, and 8 bit A/D minimum must be used. An 8 bit A/D will have a 48 dB signal-to-quantization noise at best.

We finally decided to use the ADS7810/19 A/D from Burr Brown. The ADS7810 accepts a ±10 volt maximum input, and the ADS7819 will allow up to ±2 volts input. Both A/Ds are 12 bit successive approximation converters, have a minimum 68 dB CNR, and has an 800 Ksps rate. Burr Brown provides a low cost evaluation module, the DEM-ADS7810/19, for the evaluation and implementation of applications using either of the A/Ds. The A/D evaluation board contains signal conditioners, an on board sample rate clock, and connectors for interfacing with external hardware.

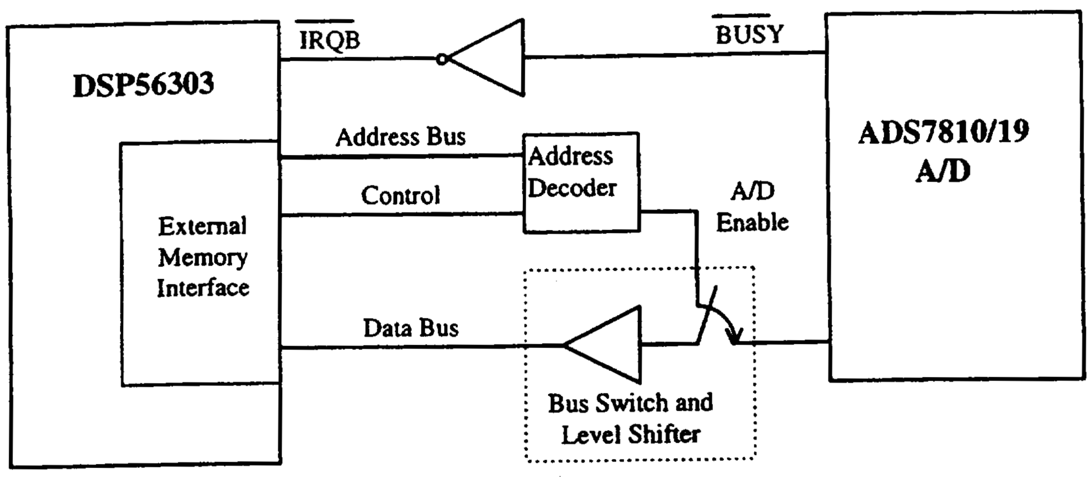


Figure 2-1: DSP A/D Interface Block Diagram

3. A/D DSP Interface Design

Interfacing the DEM-ADS7810/19 A/D evaluation board to the DSP56303EVM is rather straight forward requiring only two major hurdles to be overcome: address decoding and signal conditioning.

3.1. *Triggering DSP Interrupt Exceptions*

In many DSP applications, some events may occur that demand immediate action on the part of the DSP. In our specific application, we must insure that the DSP will read the A/D sample, when the A/D has completed a conversion. To accomplish this, we can have the A/D trigger the DSP to interrupt its current program execution, and read the sample from the A/D. After the DSP has read the data sample, it can return to its normal execution where it left off.

The DSP56300 core can implement up to 128 types of interrupts. However, the number of interrupts implemented on a specific DSP in the 56300 family depends entirely on the number and type of peripherals designed into that DSP. The DSP56303, for example, has 42 interrupts. Detailed information on the interrupt configuration of the DSP56303 can be found in Chapter 4, *Core Configuration*, of the DSP56303 User's Manual.

Associated with the interrupts is the vector table. The vector table is a 256 word table located in program memory. The base of the vector table is configurable with the Vector Base Register (VBR) that points to the lowest word in the vector table. The vector table must begin on a 256 word boundary, i.e., the least significant byte of the address must be zero. The table consists of 128 entries of two program words each. These two program words can be either two one-word assembly instructions, or a single two-word assembly instruction. Each interrupt is associated with a specific entry within this table. A portion of the DSP56303 vector table is shown in Table 3-1.

When an interrupt occurs, the DSP will execute the instructions located in the two program word entry at a specific offset from the VBR for the corresponding interrupt. For example, if an IRQA event occurs, the DSP will execute the instructions located at P:VBR+\$10 and P:VBR+\$11. The DSP then automatically returns execution to the location where if left off from. This is an example of a fast interrupt, since it is only one or two instructions in duration.

Interrupt Starting Address	Interrupt Source
VBR+\$00	RESET
VBR+\$02	Stack Error
VBR+\$04	Illegal Instruction
VBR+\$06	Debug Request
VBR+\$08	TRAP
VBR+\$0A	Non-maskable Interrupt (NMI)
VBR+\$0C	(reserved)
VBR+\$0E	(reserved)
VBR+\$10	IRQA
VBR+\$12	IRQB
VBR+\$14	IRQC
VBR+\$16	IRQD
VBR+\$18	DMA channel 0
VBR+\$1A	DMA channel 1
VBR+\$1C	DMA channel 2
VBR+\$1E	DMA channel 3
VBR+\$20	DMA channel 4
VBR+\$22	DMA channel 5
VBR+\$24	Peripheral interrupt 1
M	M
VBR+\$FE	Peripheral Interrupt 110

Table 3-1: DSP56300 Interrupt Vector Table

At times, an interrupt service routine (ISR) of two instructions may not be enough to completely service the interrupt and a longer ISR is required. In this case, the interrupt's two word entry into the vector table may be a jump to subroutine (JSR) to a much longer ISR. A long interrupt ISR may be of any length. However, the DSP does not automatically return to normal execution where it left from at the conclusion of the ISR. Therefore, a long interrupt ISR must end with a return from interrupt (RTI) instruction.

The DSP56303 has four external interrupt input pins which may be used for external triggering of interrupts. These interrupts are IRQA\, IRQB\, IRQC\, and IRQD\. On the DSP56303EVM, IRQA\ and IRQD\ are each connected to a push-button switch to allow programs to be written to react to a user pushing the buttons, such as selecting various filters, for example. Interrupts IRQB\ and IRQC\ are not connected to anything on the board, and the signals are also available on an external connector J8. Therefore, we decided to use IRQB as an input to trigger an interrupt when the A/D has completed a sample conversion.

The ADS7810/19 A/D BUSY\ signal can be used to trigger the DSP IQRB\ interrupt. BUSY\ is an asserted low signal that is low while the A/D is in the conversion process. When the A/D has finished a conversion, and is driving the output sample on it's data pins, BUSY\ is transitioned to a logical high state.

The DSP56300 external IRQ interrupts can also be programmed to trigger interrupts on one of two events: a negative going edge, or an asserted low level. Therefore the IRQB\ signal can be derived from the A/D BUSY\ signal through an inverter. The A/D BUSY\ signal will go high and become de-asserted at the instant a sample conversion has finished, which, will cause the IRQB\ signal to go from a high to a low state, asserting IRQB\. An IRQB\ interrupt can then be configured to trigger on a negative going edge, as illustrated in Figure 3-1. This will cause a DSP IRQB\ interrupt immediately after the sample conversion is completed, with minimal delay due to the inverter.

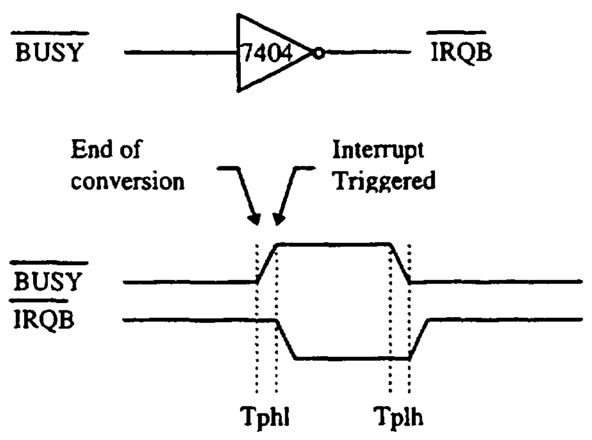


Figure 3-1: DSP IRQB signal generation from A/D BUSY signal showing inverter propagation delay

3.2. *DSP Peripheral I/O memory space*

The DSP56300 architecture consists of three separate memory spaces: one space for program and two memory spaces for data memory. The data memory spaces are called X and Y memory. The program memory is referred to as P memory. This type of microprocessor architecture with separate memory spaces is referred to as a Harvard architecture. Each of the P, X, and Y memory spaces are 2^{24}x24 bits, or 16 mega-words, in size. A diagram representing the DSP56303 memory spaces and their organization is presented in Figure 3-2.

The P memory space is used for the storage of the executable program code (instruction opcodes, operands, etc.). The two data memory spaces, X and Y, are used for the storage of any data required by the executable, such as filter coefficients, data samples, variables, constants, etc.

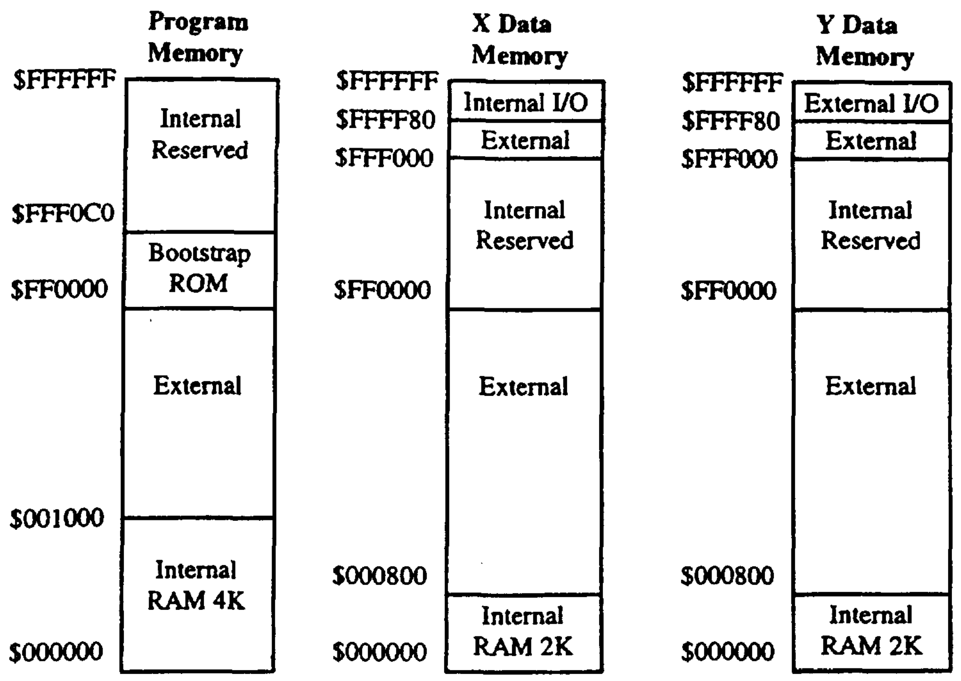


Figure 3-2: DSP56303 Default Memory Space Configuration

The X and Y data memory spaces also contain the memory mapped registers for peripherals. The upper 128 words ($FFFF80-$FFFFFF) in the X and Y data RAM of the DSP is reserved for I/O peripheral addressing. Internal peripherals (such as the timers, serial communications interface, and PLL, etc.) are mapped into the upper 128 words of the X data space. The organization of the X memory internal peripheral register mapping is defined by the specific DSP, and described in the corresponding DSP's User's Manual in Chapter 3, *Memory Configuration*. The upper 128 words of the Y data space, however, is reserved for external peripherals connected to the DSP by the user.

3.2.1. Short Addressing Advantages to I/O Peripheral Registers

This upper block of 128 words in each the X and Y data spaces can be addressed in a short addressing mode with bit manipulation and move peripheral (MOVEP) instructions. Normally, in long addressing, each instruction opcode is followed by an operand. The operand contains the entire 24-bit address required for the instruction. Long addressing modes require two 24-bit words to execute: one for the instruction opcode, and one for the instruction operand, the address. The short addressing mode allows for addressing the I/O peripherals by partially encoding the address within the instruction opcode. In I/O short addressing, 7 of the opcode bits are used to contain the lower 7 bits of the data memory peripheral address required for the instruction to execute. The other 17 bits of the data memory address, are assumed to be a specific default value. In the case of I/O short addressing, the upper 17 bits of the data address is assumed to be $FFFF80. For more information see Page A-185 of the 56300 Family Manual.

Long Addressing:
MOVE X:$FFFFA5,X0

0 1 0 0	0 1 0 0	0 1 1 1	0 0 0 0	0 0 0 0	0 0 0 0	Opcode
1 1 1 1	1 1 1 1	1 1 1 1	1 1 1 1	1 0 1 0	0 1 0 1	Operand

Short Addressing:
MOVEP X:$FFFFA5,X0

0 0 0 0	0 1 0 0	0 1 1 1	0 0 0 0	1 1 0 0	0 1 0 1	Opcode

Figure 3-3: Example of long and short address mode opcodes

An example of I/O long and short addressing is presented in Figure 3-3. In each case, the contents of a peripheral register located at X:$FFFFA5 is read into the X0 register.

The implementation of a short addressing mode to access the I/O peripheral memory is important as it allows moves from one memory location to another location in memory to be performed, as is commonly done when sending or receiving data through peripherals. With long addressing alone, this would not be possible with the DSP56300 instruction set. Instead, a long move from memory into a register followed by a second long move from the register to memory would have to be performed.

3.3. DSP External Memory Interface

The DSP56300 family contains a very flexible external bus expansion interface. Although the DSP56303, and other DSP56300 family DSPs, do contain internal RAM, the user may often wish to greatly add more memory or other peripherals external to the DSP. The DSP56300 is capable of glue-less interfacing with static or dynamic RAMs of various speeds.

The DSP56300 family DSP has a 24 bit wide address bus that allows it to access a total of 16Mx24 in each Program and X and Y data space. However, only the lower 18 address lines are available external to the DSP56303 through the external memory interface. This is due to limiting of the number of external pins to keep a small package size. This apparent short coming is not a problem as the DSP56300 family includes four external signals that can be configured to decode the twelve most significant address lines internally. These Address Attribute signals allow the user to map memory or other peripherals into the full 16Mx24 memory space of the DSP56303, although only 18 address lines are available external to the DSP. In addition, the address attribute signals also greatly reduce the complexity often needed to perform address decoding when mapping external devices to the DSP.

The user is allowed great flexibility in configuring the address attribute signals to perform custom address decoding internal to the DSP in the external memory interface. This customization includes setting the number and value of the most significant address bits to compare, specifying which of the three memory spaces (P, X or Y), the type of memory access to perform, and even the assertion level of the address attribute signal.

The DSP56303EVM has utilized two of the four address attribute signals for external RAM. Address Attribute 0 (AA0) is used for the connection of external fast SRAM. The EVM also includes a Flash Programmable/Erasable Read Only Memory (PEROM) in which the DSP program code can be programmed into. This allows the EVM to boot externally from the flash PEROM for use in stand alone operation. The Flash PEROM is interfaced with the DSP56303 by using the Address Attribute 1 (AA1) signal. The third address attribute signal, AA3, originally was not used in the DSP56303EVM, but was reserved also for use with interfacing with the external SRAM. The Address Attribute 3 (AA3) signal is intended to be used to map the external SRAM into the two DSP data spaces, X and Y, with later DSP56303 silicon that allowed this function. By using only AA2 to map the external SRAM, the memory would be mapped as one unified memory chunk. The only address attribute signal not used or reserved on the DSP56303EVM is AA2. It was therefore decided that this signal would be used to decode the higher 12 bits of the address mapping for the A/D evaluation board.

The decision was made to map the A/D evaluation board into the external peripheral I/O memory space in Y memory in order to take advantage of the short I/O addressing mode the DSP56300 provides. This would require the A/D evaluation board to be mapped into the memory range of Y:$FFFF80-$FFFFFF in which the most significant 12 address bits would always be set. We realized that the nature of this project may find new requirements within the DAMA project, or perhaps a new role in another project requiring a high speed A/D. Therefore the decision was made to decode a few of the least significant address bits on the A/D interface board. If the AA2 signal were as the only qualification to enable the A/D evaluation board, then the A/D would be mapped into the Y data memory at any time the range of Y:$FFF000-$FFFFFF were addressed. Although this would certainly allow the A/D board to be addressed within the external I/O range of Y memory, it would preclude the addition of other external hardware that may be required in the future. Therefore, a few of the least significant address bits would be decoded on the A/D board and qualified with the AA2 signal. This would cause the A/D evaluation board to be mapped into specific sections of the Y:$FFF000-$FFFFFF region.

Instead of having the A/D interface board mapped into the entire 4K region of Y:$FFF000-$FFFFFF, the least significant external address lines can be decoded. The 74688 logic device was identified to accomplish this task. The '688 compares two 8 bit inputs, Q0-7 and P0-7, and if they are equal, then the '688 will assert it's output (P=Q\), if enabled. Since we wish to enable the A/D when AA2 is asserted low during a DSP external read, two of the eight Q inputs were used for these signals. The corresponding P inputs are connected to ground. The AA2 signal and RD\ signals connected to the Q inputs are then compared to logic low values applied to the P inputs. The remaining six Q0-5 inputs are connected the lower six external address lines, A0-5. These six address lines are compared to values applied to the P0-5 inputs. To allow the user flexibility in changing the location in Y memory which the A/D interface board is mapped into, the corresponding P0-5 inputs are connected to small switches that select whether the input is connected to +5 volts, or ground. When the switch is open, the input is connected to ground through a 10K ohm pull-down resistor. If the switch is closed, the input is connected directly to +5 volts providing a logical high.

By qualifying the 6 least significant address lines, the memory block from Y:$FFF000-$FFFFFF is broken up into 64 blocks of 64 words that is reflected into each of the blocks. The A/D interface board will be mapped into a single location in each 64 word block. For example, if all the switches are open, and each of the address lines A0-5 are compared to a logic low, then the A/D board will be mapped into the location Y:$FFF000+$40·n, where n is an integer from 0 to 63. One of these locations in which the A/D board will be mapped into is Y:$FFFFC0, which is inside the external I/O peripheral memory space that can be accessed using short I/O addressing.

Figure 3-4 shows how the 24 address bits of the DSP56303 would be qualified to address the A/D evaluation board. The most significant 12 bits are qualified internally to the DSP56303 by the external memory interface logic as AA2, as configured in the AA2 configuration register AAR2. The least significant 6 bits of the address bus is qualified by the 74688 8 bit magnitude comparator and the DIP switches set by the user on the interface board. The remaining 6 address bits, A6-A11 are considered "don't cares" and are not qualified in the mapping of the A/D evaluation board into the DSP's external memory space.

A23	A22	A21	A20	A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
1	1	1	1	1	1	1	1	1	1	1	1	X	X	X	X	X	X	S5	S4	S3	S2	S1	S0

1 - Twelve most significant bits qualified by AA2

X - Don't Cares. These address bits are not qualified in mapping the A/D into memory

S5-S0 - Address bits that are qualified by the 74688 magnitude comparator and DIP switch settings on the interface board.

Figure 3-4: A/D Interface board Address Mapping

3.4. A/D to DSP signal level conversion

The Burr Brown ADS7810/19 A/D is a +5 volt logic device. The Burr Brown technical specifications for the A/D show that the digital output high voltage will be a minimum of +2.8 volts, but will probably be as high as the digital supply voltage. By measuring the logic levels of the A/D's digital data outputs, it was shown that they will produce levels comparable to the +5V digital supply.

The DSP56303, however, is a low voltage device with a supply voltage rating of V_{CC}=3.0-3.6 volts. Only specific DSP I/O pins, which are referred to as *5 volt tolerant*, are capable of being driven at levels greater than this supply voltage. These specific pins are able to accept input voltages up to a maximum of V_{CC}+3.95 volts, or 7.25 volts. One of these 5 volt tolerant pins is the IRQB\ pin, which is used to interrupt the DSP at the end of an A/D sample conversion. The external memory interface data bus is among the other signals that are not 5 volt tolerant. According to the DSP56303 Technical Data Sheets, the voltage applied to the data bus pins should not exceed the DSP supply voltage of 3.0-3.6 volts. Since the A/D drives the data bus at a higher voltage than the DSP data bus can handle, a level conversion must be performed. This is a common problem today in applications where there is a mix of older, higher voltage technology, and newer low power systems that operate at 3.3 volts, or even as low as 1.8 volts.

The Quick Switch QS3384 by Quality Semiconductor, Inc. provides a solution that introduces minimal propagation delay to the signals with a minimum of glue components. The QS3384 will act as a switch, allowing the A/D to drive the DSP data bus when so enabled, in addition to providing level translation. The QS3384 is essentially an integrated circuit consisting of ten N-channel Field Effect Transistors (N-FET), whose gates are connected to one of two common enable signals. Each enable signal is used to turn on or off five of the NFET transistors at a time. A schematic of one half of the QS3384 is shown in [TB1]Figure 3-5.

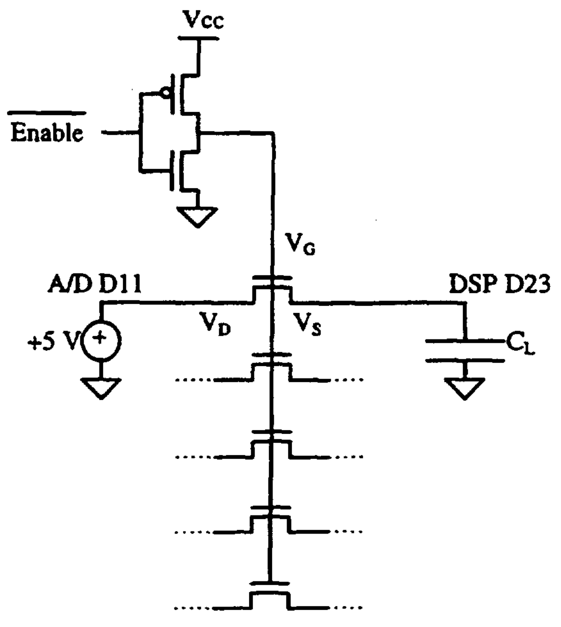


Figure 3-5: Schematic Diagram of one-half of the QS3384 10-bit Bus Switch

Quality Semiconductor publishes two application notes (TN-07 and AN-11a) that describe the level translation feature of the QS3384 bus switch. When the switch ENABLE signal is low, the P channel FET (P-FET) of the inverter structure conducts. This applies the V_{CC} voltage to the gate of each of the five bus switch N-FETs. When the gate voltage, V_G, of the N-FET becomes larger than the threshold voltage (V_t) of approximately 1 volt, the N-FET will conduct. When the A/D is driving the N-FET with a logic high signal of +5 volts, then it can be considered the drain of the N-FET. The N-FET source will then be connected to the DSP data which will appear as a capacitive load, C_L, to the N-FET. With the N-FET conducting, it will begin to charge the data input C_L, until the N-FET gate to source voltage, V_{GS}, is equal to the threshold voltage, V_t. By applying Kirchoff's Voltage Law, one can then determine the maximum source voltage, V_s, that will be applied to the DSP data input pin (equation 2).

$$V_s = V_{CC} - V_t \qquad (2)$$

The N-FET threshold voltage is typically about one volt. Therefore, the maximum the QS3384 can drive the DSP data input is $V_s = 5V - 1V = 4V$. However, four volts still exceeds the maximum input voltage allowed on the DSP data input pins. This can be solved by lowering the Vcc voltage applied to the QS3384. A forward biased diode will produce approximately a 0.7 V drop. By placing the forward biased diode in series between the A/D interface board's 5V supply voltage and the QS3384 V_{CC} pin, the voltage supplied to the QS3384 will be approximately 4.3 volts. With V_{CC} now at 4.3 volts on the QS3384, we can see from equation (2) again that the maximum voltage at which the QS3384 will drive the DSP data input will be V_S=5V - 0.7V - 1.0V = 3.3V.

This provides an input logic high voltage that will not exceed the specifications for the DSP.

By enabling the QS3384 and turning on the N-FET switch when the A/D is driving a logic low in the data pin, the A/D will discharge the DSP data pin (C_L). This will cause a logic low level on the DSP input pin.

The QS3384 enable signal is generated by the DSP's Address Attribute AA2 signal. Since AA2 will be configured only to assert on external Y memory reads, the QS3384 will never be enabled (N-FET switches 'on') when the DSP is driving the data bus. With the QS3384 enable signal at a logic high, the N-FET switches will not conduct, providing a high impedance (few mega-ohms) between the A/D and DSP data busses. The propagation delay between the QS3384 enable signal going low, and the N-FET switches conducting is less than 5 nS.

3.5. Data Bus Connections

We wish to represent the sampled signal with maximum dynamic range within the DSP. Therefore, the binary word representing the A/D sample should be left justified when transferred onto the DSP's data bus. In addition, we must insure that the sign bit of the A/D will be correctly stored in bit 23 of the DSP's data registers. Therefore, we cannot simply connect the A/D D11 signal (A/D sign bit) to D11 on the DSP, A/D D10 to the DSP D10, etc., down to A/D D0 to the DSP D0.

To gain maximum dynamic range of the A/D samples within the DSP, the A/D data signals are connected to the higher order DSP data lines. The A/D data signals AD_D0-AD_D11 are connected to the DSP data bus as D12-D23. The sign bit of the A/D data bus, A/D D11 is connected to the DSP D23 pin. The remaining DSP data signals, D0-D11, must be a logic low when the DSP reads from the A/D to avoid corrupting the data sample and introducing noise. This must be done, as the DSP56303EVM leaves all data signals floating, if no device is driving them. Therefore, these data signals are pulled down to ground through a 10K ohm resister to insure they read as a logic zero.

4. Construction

4.1. Construction Technique

The construction technique chosen to construct the A/D interface board is wire wrapping since it can be used to easily construct the circuit without solder. Circuit changes can be easily made with wire wrapping, and at a fraction of the cost of fabricating a printed circuit board. Wire wrapping is typically used in prototype circuits, such as the A/D interface board, but can be well suited for permanent circuits as well. Wire wrapping is also suitable for medium speed digital circuits.

Wire wrapping consists of tightly wrapping about an inch of stripped no. 30 gauge insulated wire around a square post with sharp corners. Early during the construction of the A/D interface board, it was noted that the first wrap of the wire tended to loosen and pull away from the post. The wire, as it is tightly wrapped, digs into the corners of the post to make a secure connection. To prevent the first wrap from shorting the connection to nearby connections, a modified wrap method was used in which an attempt was made to insure the first one or two turns of the wire on the post consisted of insulated wire.

4.2. Layout

The layout of the A/D Interface board was developed through a trial and error method. The goal was to create a small, compact, and uncluttered board, that allowed easy cable connections to the DSP EVM board and the A/D evaluation board. Finally, the layout shown in Figure 4-1 was decided upon. The dotted lines roughly show the layout of the cables interconnecting the three boards.

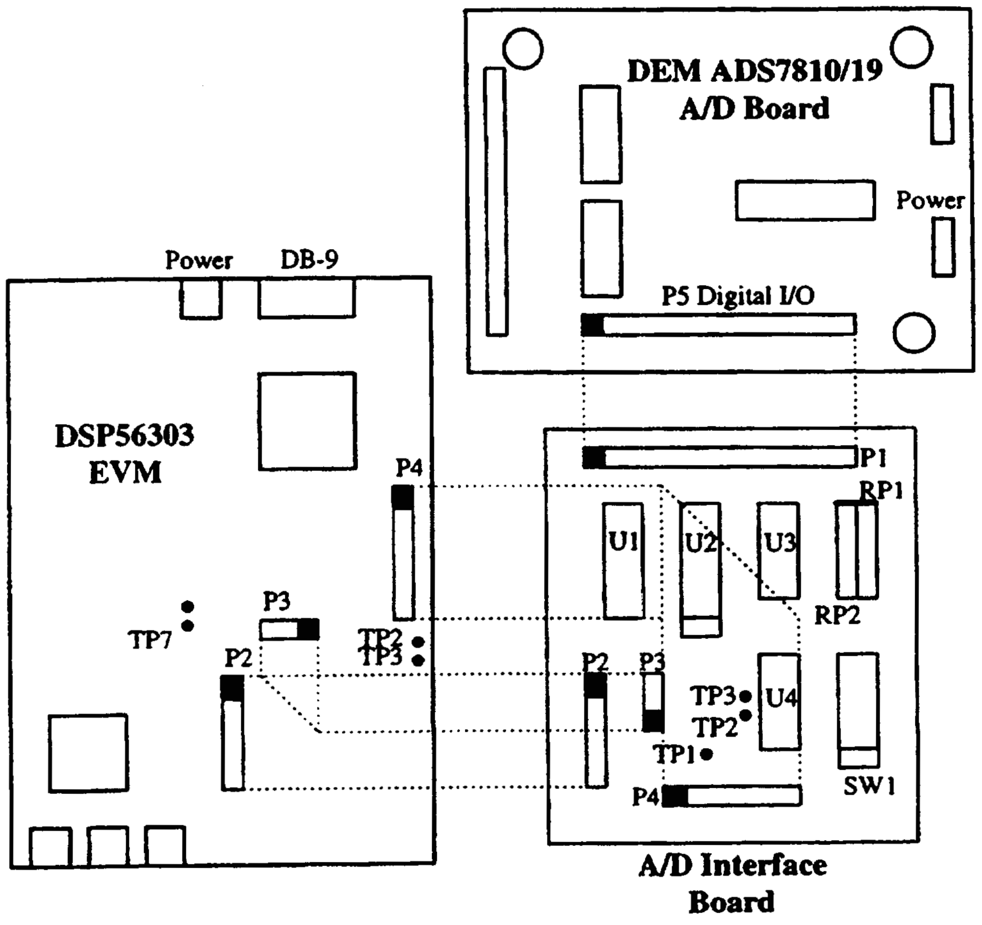


Figure 4-1: A/D Interface Layout, and Cable Connections to the DSP and A/D Evaluation Boards

5. DSP Software Drivers

To develop the driver and demo software to use the A/D interface, we take a modular approach. We begin by first examining and writing the assembly code to configure each of the various aspects of the DSP required to use the A/D. The modular code is then used as building blocks to create a sample application program to use the A/D interface. A flowchart for the DSP A/D test program is presented in Figure 5-1.

5.1. Interrupts (levels, masking, etc.)

The 56303 DSP has three levels of interrupts: 0 through 3. Level 0 interrupts are the lowest priority, while level 3 interrupts are the highest. A higher priority interrupt can interrupt a lower priority interrupt during execution. Level 3 interrupts cannot be disabled, or *masked*. Such interrupts are Hardware Reset, Stack Errors, Illegal Instruction, TRAP, etc. The other three interrupt levels, 0 through 2, can be masked off and disabled. The masking level 'threshold' is set by the I1 and I0 bits in the Status Register, SR. When the I bits in the SR are programmed to 01, interrupt levels 1 and above (1, 2, and 3) are enabled, but level 0 is masked. To enable interrupt levels 2 and 3, but mask levels 0 and 1, the I bits in the Status Register would be programmed to 10.

Peripheral interrupts can be assigned a specific priority level by programming the appropriate bits in the Interrupt Priority Register IPR-P. The DSP core interrupts are assigned a priority level by programming the IPR-C register. See the Programming Reference Sheets for details of the IPR-C and IPR-P registers in Appendix D of the 56303 User's Manual.

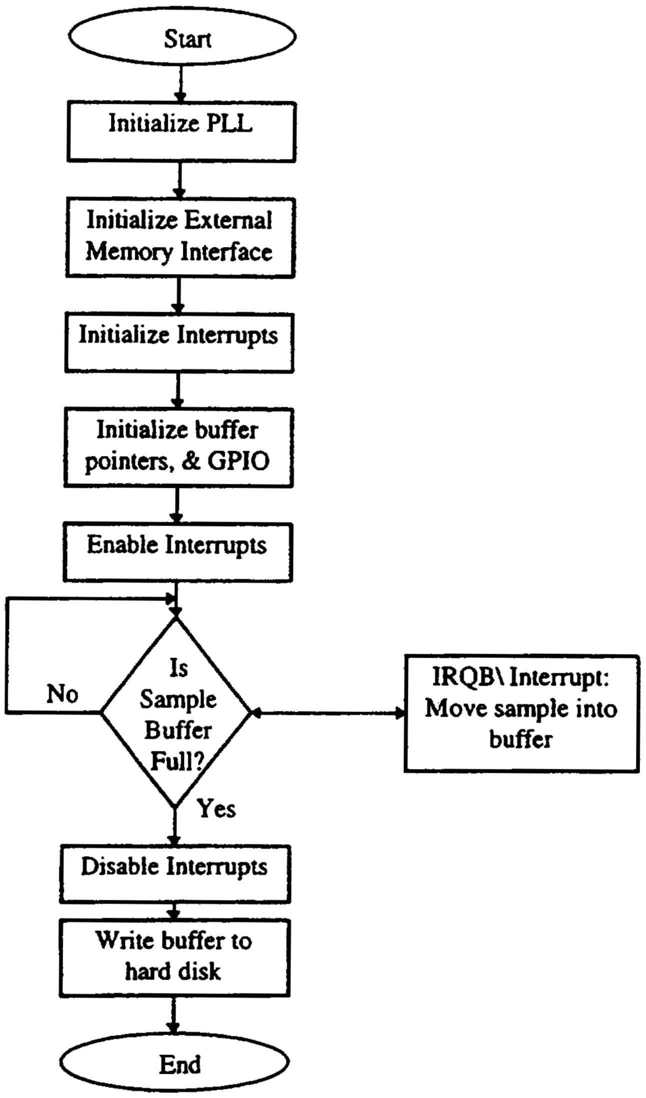

Start
Initialize PLL
Initialize External Memory Interface
Initialize Interrupts
Initialize buffer pointers, & GPIO
Enable Interrupts
Is Sample Buffer Full?
No
Yes
IRQB\ Interrupt: Move sample into buffer
Disable Interrupts
Write buffer to hard disk
End

Figure 5-1: DSP Test Program Flow

The DAMA doppler carrier tracking project requires a constant audio carrier to be generated during times that high speed A/D samples may be buffered. The audio carrier will be generated using a long ESSI interrupt. Since the high speed A/D must be serviced nearly 50 times faster than the audio carrier ESSI interrupt, the A/D interrupt should be assigned a higher priority. This will allow the A/D interrupts to occur, and samples to be buffered, without delay and even while ESSI interrupts are executing generating audio carrier output samples. The A/D IRQB\ interrupt should therefore be programmed to an interrupt priority level of 2. Only major system interrupts such as RESET, stack errors, or illegal instructions, (level 3 interrupts) will interrupt the A/D sample buffering. The ESSI transmit interrupt, and other interrupts that may be required for this project, can be configured for the lower priority levels 0 and 1.

To disable any interrupt altogether, without masking that interrupts priority level, 00 can be programmed into the respective interrupt's priority level field in the interrupt priority control register. This does not set the interrupts priority level to 0, but instead disables it completely. This method should be used when buffering of the high speed A/D samples is not required, and other peripheral interrupts of lower priority must remain enabled. Simply masking the priority level 3 interrupts in the Status Register will not achieve the required results as it also disables the lower interrupt priority levels, such as the ESSI interrupts which will be used for the audio CODEC on the DSP56303EVM.

The DSP56300 external IRQ interrupts can be programmed to trigger interrupts in one of two modes: a negative going edge, or a low level. Since IRQB\ will have a negative edge derived from the A/D's BUSY\ signal through an inverter, the IRQB\ interrupt should be generated triggering on a negative going edge.

```
; CORE Interrupt Priority and Configuration
IBM    equ    1            ;IRQB trigger (0 level, 1 negative edge)
IBP    equ    2            ;IRQB priority level 0 (low), 1, or 2 (high)

IBL    equ    (IBM<<2)+(IBP+1)&3      ;Define IRQB part of IPR-C register

IPRC   equ    IBL<<M_IBL0&M_IBL ;Define Core Interrupt Priority Reg value
       movep  #IPRC,x:M_IPRC    ;Initialize Interrupt priority/config

;To enable all interrupt levels
       andi   #$FC,MR           ;Enable all interrupts (levels 0-3),
                                ;I1:0 in Mode Reg

;To disable all interrupt levels (0-2)
       ori    #$02,MR           ;Disable interrupt levels 0&1,
                                ;leave level 2 (IRQB) enabled
```

```
IBP   equ   2               ;IRQB priority level 0 (low), 1, or 2 (high)

IBL   equ   ([illegible]M<<2)+(IBP+1)&3      ;Define IRQB field of IPR-C register

IPRC  equ   IBL<<M_IBL0&M_IBL ;Define Core Interrupt Priority Reg value
      movep #IPRC,x:M_IPRC    ;Initialize Interrupt priority/config

;To enable all interrupt levels
      andi  #$FC,MR           ;Enable all interrupts (levels 0-3),
                              ;I1:0 in Mode Reg

;To disable all interrupt levels (0-2)
      ori   #$02,MR           ;Disable interrupt levels 0&1,
                              ;leave level 2 (IRQB) enabled
```

Listing 5-1: Configuring, enabling, and disabling DSP core interrupts

5.2. *Address Attributes and Memory Configuration*

As previously mentioned, the A/D interface board is mapped into the DSP's external I/O peripheral space in Y data memory. The address attribute (AA2) pin is used to decode the upper 12 address pins and memory space within the DSP's external memory interface. Configuring the AA2 pin to decode the address and memory space to properly enable the A/D interface is performed by programming the Address Attribute Register (AAR2). The AAR2 register, which is described in detail in the DSP56300 Family Manual, is a 24 bit memory mapped register (in internal X I/O memory) with nine bit fields that control the addressing conditions under which the AA2 pin will be asserted, and various functions the external memory interface will perform when the AA2 pin is asserted. These fields within the AAR2 register, and the mode in which they are programmed, are listed in the table below:

Field	Num bits	bits #'s	Value	Setting	Description
BAC0-11	12	12-23	$FFF	$FFFxxx	Address to Compare
BNC0-3	4	8-11	$C	12	Number of Address bits to compare
BPAC	1	7	0	Disabled	Byte packing enable
BAM	1	6	0	Disabled	Address Muxing
BYEN	1	5	1	Enabled	Y data space enable
BXEN	1	4	0	Disabled	X data space enable
BPEN	1	3	0	Disabled	Program space enable
BAAP	1	2	0	Active Low	AA pin polarity
BAT0-1	2	0-1	$1	SRAM	External access type

Table 5-1: Address Attribute 2 Register (AAR2) settings for the A/D Interface Board

The first field, BAC0-11, is twelve bits which describes the bit pattern that will be matched to the state of address bus lines A12-23 as one of the criteria for asserting the address attribute pin. The BNC0-3 field is a four bit field that designates how many of the upper address bus lines (and the BAC0-11 bits) are to be matched with the address to compare field. When this field is programmed to $C, all twelve BAC0-11 bits are matched to the address bus lines A12-23. If A12-23 are equal to the settings in BAC0-11, and other conditions described below are met, then the corresponding address attribute line will be asserted. Fewer address compare bits may be matched to the address bus, but the most significant address bus lines are always the signals compared. The byte packing option (BPAC) is for interfacing the DSP to byte wide memories, which will not be the case with the A/D interface. The address muxing (BAM) allows the least significant address lines, A7-0, to be driven externally on the higher address pins which are not needed externally as they are decoded by matching the address to compare field. This allows the connection of devices to the higher order address lines to reduce capacitive loading on the lower address lines in large applications. For the A/D interface, the address muxing option is not needed. The next three options specify the memory spaces in which the address attribute signal will be asserted. If the X enable (BXEN) option is set, then the address attribute signal will be asserted for X memory accessed, within the address space specified by the BAC0-11 and BNC0-3 bit fields previously described. The Y memory space is enabled with the BYEN bit, and the program memory space is enabled by setting the BPEN bit. The address attribute signal may be configured to assert on accesses to one, two or all three of the memory spaces. The polarity of the address attribute is controlled by the BAAP bit. The A/D Interface board was designed such that the A/D would drive the DSP bus with data when AA2 is asserted low, therefore this bit is cleared. The final field, BAT0-1, specify the type of external memory interface control signals are required for the memory space being addressed by this address attribute. The DSP56300 is capable of interfacing with other devices, such as Synchronous SRAM, and Dynamic RAM, however, the A/D interface board is designed to appear as static RAM (SRAM).

The DSP56303EVM contains a 32K-word external SRAM that is connected to address attribute 0 (AA0). The EVM User's Manual, contains a description and example for programming the AA0 to use the external SRAM as a unified (X and Y) memory mapped at XY:$010000-$017FFF. This portion of the memory space is not occupied by any other devices, and there is no reason to deviate from the settings suggested in the EVM User's Manual.

A portion of the assembly code to configure the address attribute registers is shown in Listing 5-2.

```
AAR0          equ    $010931       ;Unified X/Y SRAM 32K words
                                   ;$010000-$017fff, no byte packing, X and Y
                                   ;enable, no address MUX
AAR2          equ    $fffc21       ;Compare upper 12 address bits to $FFFxxx,
                                   ;no byte packing, no address MUX,
                                   ;Y enable, SRAM access
```

```
M_AAR0       equ    $FFFFF9              ; Address Attribute Register 0
M_AAR2       equ    $FFFFF7              ; Address Attribute Register 2
             movep    #AAR0,x:M_AAR0    ;Initialize AA0 to ext SRAM
             movep    #AAR2,x:M_AAR2    ;Initialize AA2 to A/D Board
```

Listing 5-2: Configuring the Address Attribute Registers

5.3. External Bus Wait States

The DSP56300 EMI can be programmed to insert a number of extra clock cycles during external memory accesses called *wait states*. These wait states are inserted to allow slow memories extra time to complete memory reads or writes. By programming the Bus Control Register (BCR), the number of wait states inserted for external access can be programmed for each of the areas mapped by address attributes 0-3, and for external accesses that are not mapped to an address attribute (default). On reset, or power-up, the BCR defaults to provide the maximum number of wait states, either 7 or 31. This guarantees that the DSP has access on reset to even the slowest of memories that the user may connect to the DSP. This is especially critical for applications in which the DSP boots from reset executing application code from the flash PEROM on the EVM. Once the DSP is executing the program in flash, the program can lower the number of wait states used to access external memory to make the program execution more efficient.

The number of wait states required for the DSP to read from the A/D interface board was first estimated by determining how much propagation delay the A/D interface board address decoding would be. Once the estimated delay was known, the number of wait states required for the DSP to insert when operating at a clock rate of 80 MHz was determined from the DSP56303 technical data sheets. When the A/D board was built and tested, it was shown to operate reliably with 6 wait states.

The number of external EVM SRAM wait states was determined in a similar manner. The number of DSP wait states required to access the external 15 nS SRAM was calculated from the DSP read and write speeds from the DSP technical data at a DSP operating clock 80 MHz. The EVM was shown to read and write to external SRAM reliably with the A/D interface board connected with 3 wait states.

```
;Wait State Settings
DEFAULT_WS   equ   15                 ;default are wait states (0-31)
SRAM_WS      equ   03                 ;32KW SRAM (0-31)
FLASH_WS     equ   00                 ;FLASH (0-31)
PERIPH_WS    equ   06                 ;A/D Peripheral board (0-7)


AREA0        equ   SRAM_WS            ;AA0 wait states (0-31)
AREA1        equ   FLASH_WS           ;AA1 wait states (0-31)
AREA2        equ   PERIPH_WS          ;AA2 wait states (0-7)
AREA3        equ   DEFAULT_WS         ;AA3 wait states (0-7)


BBS          equ   0                  ;Bus State
BLH          equ   0                  ;Bus Lock Hold
BRH          equ   0                  ;Bus Request Hold


BCR          equ   (BBS<<21)+(BLH<<22)+(BRH<<23)+\
(DEFAULT_WS<<16&M_BDFW)+(AREA3<<13&M_BA3W)+(AREA2<<10&M_BA2W)+\
(AREA1<<5&M_BA1W)+(AREA0&M_BA0W)

M_BCR        equ   $FFFFFB            ; Bus Control Register
             movep #BCR,x:M_BCR       ;Initialize Bus Control Register
```

Listing 5-3: Configuring the Bus Control Register, and External Wait States

5.4. A/D Interrupt Service Routine

When the A/D has completed a sample conversion, it will trigger an IRQB\ interrupt on the DSP56303. The DSP will then insert the two program words located at vector table entry for the interrupt into its execution pipeline. To service the IRQB\ interrupt, the data sample is read from the A/D interface board, and stored into X data memory, pointed to by the R4 address register. Since the A/D board was mapped into the external I/O Y memory peripheral space, this can be accomplished with a single peripheral move instruction (MOVEP) using I/O short addressing. Since the short I/O move instruction will use only one word of the two word vector table entry, we must provide a valid instruction in the second word as well. This could be a counter update, to count the number of samples read by the DSP. In this case, we use the address register R4 as a counter as well as a memory pointer. Therefore, we fill the second word of the IRQB\ vector table entry with a do nothing instruction (NOP).

```
; Define Interrupt Vector Table p:$0000-$00ff
I_VEC equ    $0                ;Vector base address
      org    pli:I_VEC
      ds     $100              ;Reserve P space for Interrupt Table

      movec #I_VEC,vba         ;Configure vector base address register

      move   #AD_BUF,r4        ;Point R4 to X: buffer block
      move  #BUF_SIZ-1,m4      ;Modulo addressing

; 7819EQU.ASM - equates for the Burr Brown ADS7819 to DSP56303EVM
; interface board
BB_ADR       equ   $0                 ;DIP switch address on interface
                                      ;board for address decoder ($0-$3f)

BB7819_DR    equ   $FFFFC0+BB_ADR     ;ADS7819 Data Register

; IQRB - A/D Interrupt Service Routine
; This fast interrupt will read data from the A/D interface board
; mapped into external Y: peripheral memory space, and store the
; data sample into X: memory.
      org    pli:I_IRQB
      movep y:BB7819_DR,x:(r4)+       ;Read sample from A/D and store
      nop                             ;2nd word is no-operation
```

<u>**Listing 5-4: A/D (IRQB) Interrupt Service Routine**</u>

5.5. *General Purpose I/O*

The DSP303 provides the ability to use 34 bidirectional external pins for general purpose inputs or outputs, or as internal peripheral signal pins as defined by the user. If the user is not requiring the use of an internal peripheral, or even a specific pin of an internal peripheral, pins can be programmed as general purpose input or output (GPIO) pins. For testing of the high speed A/D interface, it was decided that the use of a general purpose output pin would be useful in providing a trigger enable for a logic analyzer. When the IRQB\ interrupt is enabled, the general purpose output pin would be asserted. When the sample buffer is full, and the IRQB\ interrupt is disabled, the general purpose output pin would be deasserted to signify that the DSP has stopped reading the A/D samples.

Each GPIO pin is associated with a specific bit in three registers. The first register is the Port Control Register. When the corresponding bit for the pin is set, then that pin is internally connected to the peripheral. If this bit is cleared, then the pin is internally disconnected from the peripheral, and the pin is configured as a GPIO signal.

The second register, the Port Direction Register, configures a GPIO pin as an input or an output. If the corresponding bit for the pin is set, then the pin will be a general purpose output pin. If this bit is cleared, then this bit will be designated as a general purpose input pin. If the pin has been configured as a peripheral pin by the Port Control Register, then the corresponding bit in the Port Direction Register is ignored.

If the GPIO pin has been programmed as an input pin, then the current logic state of the pin may be read from the corresponding bit in the Port Data Register. When programmed as an output, then the value written into the corresponding bit in the Port Data register will specify the voltage level driven out on the pin. If a zero is written, the pin will be driven low (ground). If a one is written, then the pin will be driven high (5 volts).

The DSP56303EVM has an external connector for the ESSI1/SCI peripheral pins. These peripherals are not used by the EVM, or the A/D interface board. Therefore, we can use some of the ESSI1 pins as general purpose outputs to enable the logic analyzer when the IRQB\ interrupt is enabled. The source code to configure the GPIO pin, and to change it's logic level is shown in Listing 5-5.

```
;Initialize ESSI1 SCK1 pin (Port D, pin 3) as GPIO Output
      bclr  #3,x:M_PCRD       ;SCK1 ESSI1 pin GPIO
      bclr  #3,x:M_PDRD       ;Initialize output data to low
      bset  #3,x:M_PRRD       ;Make pin an output

;Assert SCK1, then enable interrupts
      bset  #3,x:M_PDRD       ;Assert Port D pin 3 (SCK1) to enable
                              ;logic analyzer capture
      andi  #$FC,MR           ;Enable all interrupts (levels 0-3)

;Deassert SCK1, then disable interrupts
      ori   #$03,MR           ;Disable all interrupts (levels 0-2)
      bclr  #3,x:M_PDRD       ;deassert Port D pin 3 to stop logic
                              ;analyzer capture
```

Listing 5-5: Interrupt and GPIO

5.6. Writing Buffered Samples to Disk

The Motorola ADS debugging tool incorporates several features to aid in the real-time debug of simulation data. One of these features is the ability to allow the DSP to execute instructions at full speed, and stop at specified locations in the code to read a data value, or block of values, from the host computer's disk and write it into RAM using the INPUT command. The ADS can also save a memory location, or memory block of data, to disk. The debugger allows the user to accomplish this by following a short set of rules that involve loading the R0 register with a pointer to the memory location, storing X0 with the number of data values to read or write from memory, storing a value in R1 to specify P, X, or Y memory, and executing a debug command. The DEBUG instruction causes the DSP to halt execution, and returns control to the ADS debugging tool. The ADS will determine from the program counter (PC) if an INPUT or OUTPUT command has been attached to the particular DEBUG command that was executed. If so, then data is either read from memory to the associated file, for an OUTPUT command, or data from the file is stored into the DSP RAM, for an INPUT command. This feature allows the user to test algorithms executing on the DSP at full speed, while maintaining control of the input data samples, and recording the output data for analysis.

Using this method, the performance of the DSP to A/D interface can be analyzed. The test program was written to fill the 32Kx24 bit external SRAM on the DSP56303EVM with consecutive samples from the A/D. When the external SRAM was full, all 32,767 words are written to the computer's hard disk using the ADS OUTPUT command. The data samples can then loaded be into Matlab, or a spreadsheet program, for analysis.

The assembly code that initializes the DSP's registers and places the DSP into debug mode to dump the contents of the SRAM to the hard disk is listed below.

```
      move   #($010000|(BUF_SIZ-1)),x0      ;Block size and file #1 ->X0
      move   #AD_BUF,r0               ;Buffer base address ->R0
      move    #1,r1                         ;denote X memory ->R1
      FILE1 debug                     ;Output command, enter debug mode

;To attach this command to a disk file using the ADS, enter the command:
;OUTPUT #1 P:FILE1 <filename.dat> -rf -o
; <filename.dat> is the output filename
; -rf - output fractional data format
; -o - overwrite the file if exists, create if not
```

Listing 5-6: Code to output buffered samples to disk

6. Verification

To prove that the A/D board is operational, the DSP A/D test program in appendix 0 is run and two main tests are performed. The first series of tests are to prove that the digital interface between the A/D and the DSP is operating within spec. The second test is to buffer a number of samples of a signal into memory, and verify that the data read by the DSP is indeed the data sample the A/D writes to the data bus.

6.1. Interface Logic Verification

To verify that the A/D interface is correctly interfaced with the DSP and is within timing specifications of the DSP, the DSP test program in 0 is run on the DSP. A Lecroy LC574AM 4 channel digital storage oscilloscope is used to verify the logic signal timings initially calculated for the A/D interface. The Lecroy oscilloscope has a feature that allows the user to make minimum, average, and maximum time measurements between any two events on either of the 4 channel inputs. This feature was used to determine the minimum, average, and maximum timing numbers over a minimum of 1000 iterations. The resulting timing measurements are presented along with the timings determined from various technical data sheets in Appendix B.

6.2. Data sample verification with logic analyzer

The final step in verifying that the A/D interface is correctly functioning with the DSP, we must prove that the samples generated by the A/D are correctly read by the DSP and stored into the memory. This is accomplished by using a Hewlett Packard HP16550C Logic Analyzer. A logic analyzer is capable of sampling many signals on a programmable event trigger, determining the logic levels of those signals, and even grouping signals together to form binary words. The setup configuration for testing the A/D interface is shown in Table 6-1.

Pod 1 Signal	A/D Interface signal
Clock J	IRQB\
0	A/D D0
1	A/D D1
2	A/D D2
3	A/D D3
4	A/D D4
5	A/D D5
6	A/D D6
7	A/D D7
8	A/D D8
9	A/D D9
10	A/D D10
11	A/D D11

Pod 2 Signal	A/D Interface signal
Clock K	not connected
0	DSP Port D, Pin 3 (SCK1)

Table 6-1: Logic Analyzer Connections

The logic analyzer is configured to sample each of the signals on a negative edge of clock J which is connected to IRQB\. To insure that the logic analyzer begins storing A/D samples into memory when the DSP begins, the logic analyzer is set to only store samples when the DSP's Port D, pin 3, is asserted high (I.E., the logic analyzer is gated by the DSP general purpose output pin 3 on Port D). The DSP test program asserts pin 3 of Port D (SCK1) as a general purpose output pin when the DSP has enabled interrupts for the reading and storing of A/D samples. Therefore, the logic analyzer and the DSP will begin storing A/D samples at the same instant.

The logic analyzer is capable of storing 512 data samples. Therefore, only the first 512 data samples stored in the buffer by the DSP can be verified. Data samples stored by the logic analyzer are manually verified with the samples stored in the DSP SRAM using the ADS debugger.

7. Results

To demonstrate the functionality of the high speed A/D, several signals are sampled at 800 Ksps and buffered into the DSP external SRAM. Once the 32K word SRAM is full, the entire buffer is saved to the PC hard disk using the DSP debugger. The resulting data file is then loaded into Matlab for analysis. The frequency spectrum is then computed for each of the sampled signals using a 1024 point FFT with a rectangular window.

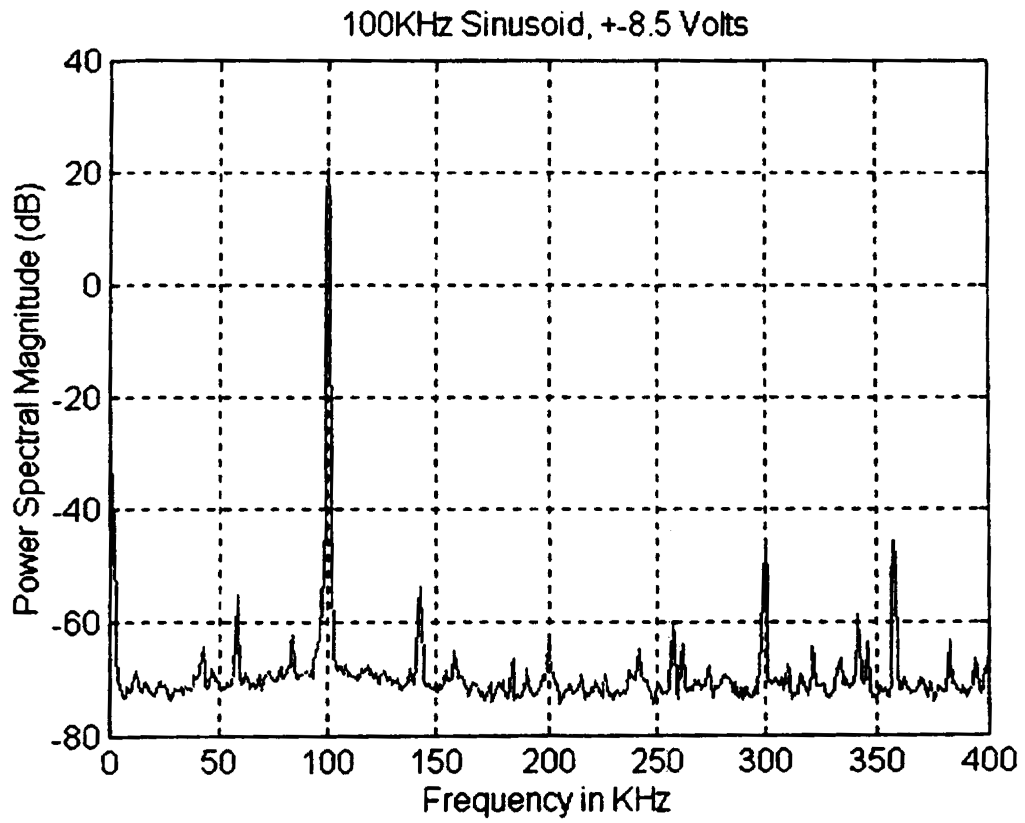


Figure 7-1: 100 KHz Sinusoid Power Spectral Density, 32767 samples, 1024 point FFT, Rectangular Window

First, a 100 KHz sinusoid was generated using a calibrated Tektronics SG5010 Programmable Oscillator. This signal was the sampled by the DSP using the high-speed A/D and interface. The resulting frequency spectrum is shown in Figure 7-1.

The second signal sampled is a 252 KHz sinusoid for which the resulting spectrum is shown in Figure 7-2. Finding calibrated sinusoidal sources for use at frequencies much above 100 KHz proved difficult. Therefore, the 252 KHz sinusoid was generated using a Hewlett Packard 8657B RF Signal Generator. The HP signal generator was configured to generate an unmodulated AM signal (raw carrier) at 252 KHz. The difference in spectral purity between the calibrated programmable oscillator and the RF signal generator is apparent when looking at the produced spectrum.

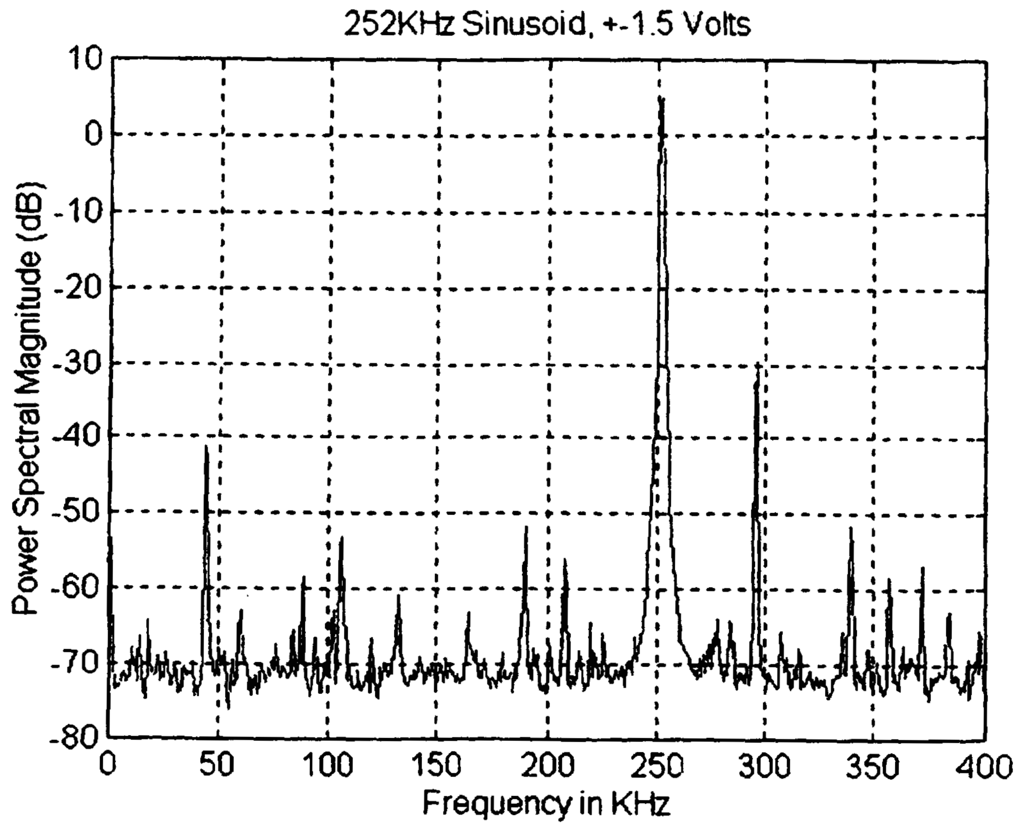


Figure 7-2: 252 KHz Sinusoid Power Spectral Density, 32767 samples, 1024 point FFT, Rectangular Window

Shown in Figure 7-3 is the frequency spectrum of a 16 KHz square wave generated by the Tektronics SG5010 Programmable Oscillator. In this figure, the odd harmonics characteristic of a square wave is clearly shown. Also shown is the aliasing of the higher order harmonics due to the lack of an anti-aliasing filter on the Burr Brown DEM-ADS7810/19 A/D evaluation board.

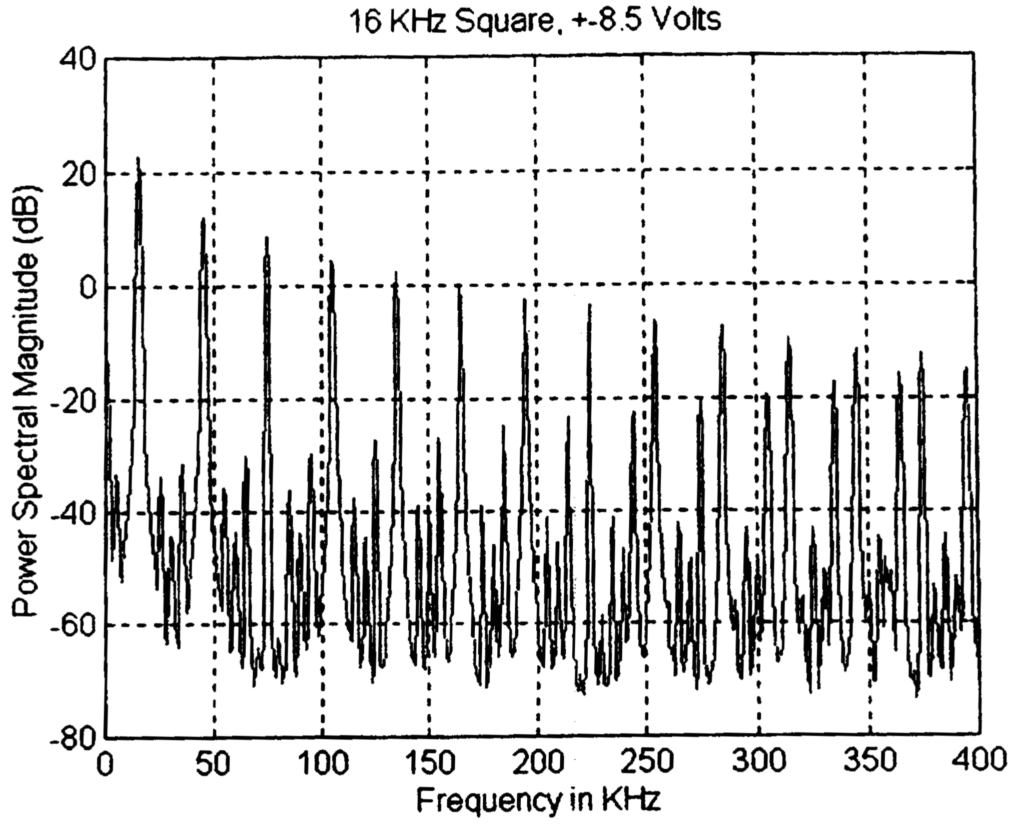


Figure 7-3: 16 KHz Square Power Spectral Density, 32767 samples, 1024 point FFT, Rectangular Window

8. Conclusion

8.1. *Conclusion from Results Observed*

As indicated from the data samples collected, it is shown that the DSP56303EVM is capable of using the interface board to reliably sampling signals at a rate of 800 Ksps. Therefore, we can accurately sample the DAMA signal and store those samples within the DSP's memory buffers for doppler detection.

8.2. *Implementation Evaluation and Concerns*

Through the course of this project, several options presented themselves which can be implemented to greatly enhance the functionality and flexibility of the high speed A/D interface board. These options should be taken into consideration in future DSP applications that require a high speed A/D.

8.2.1. Printed Circuit Board

First, the lengths of interconnecting ribbon cables between the boards, and the wire wrap methods used in the construction, tend allow for cross-talk (inductive coupling) between signals. This tends to introduce noise in the digital signals, and decrease the rate at which the digital pulses may be reliably transmitted on the interface board.

Although a wire-wrapped solution is good for a prototype board, a much cleaner solution would be to layout a printer circuit board (PCB) that would mount onto the DSP56303EVM's connectors as a daughter board. This would greatly decrease the inductive coupling between signals, and increase the operating speed of the interface.

8.2.2. Variable Sample Rates

During the course of the development and implementation of the A/D interface board, the original specifications for the DAMA project had changed. First, the limits of expected doppler shift were revised down to +- 50 KHz, from the original +- 64 KHz. In addition, it was discovered that the DAMA spreading rate of 100 Kcps would be too great, and it too was revised down to the order of 50 Kcps. This results in a new DAMA signal bandwidth of 200 KHz. If we were to continue sampling at 800 Ksps, we would be unnecessarily over-sampling the signal.

Although the specific details are beyond the scope of this report, two possible solutions could be examined to divide the sample rate of the 800 Ksps A/D interface board by an integer, *n*. A total software approach would be to take advantage of the Direct Memory Access (DMA) peripheral of the DSP56303. The DMA can be programmed to perform a transfer from one memory location, to another memory location when triggered by a programmable event. This transfer can occur with no impact to current processing of the DSP core.

A total hardware solution, might be to use a counter, such as the 74HC163, as a frequency divider. The counter would divide the rate at which the BUSY\ pulses occur by *n*, and present this signal as the DSP interrupt IRQB\.

References

[1] American Radio Relay League, *The ARRL Handbook 70th Edition*, American Radio Relay League, Newington, CT, 1993.

[2] Burr-Brown, "ADS7810," 1992.

[3] Burr-Brown, "ADS7819," 1992.

[4] Burr-Brown, "DEM-ADS7810/19C Evaluation Fixture," 1993.

[5] Chrysafis, C. and Lansdowne, S. "Fractional and Integer Arithmetic Using the DSP56000 Family of General-Purpose Digital Signal Processors," Applications Note APR3/D, Motorola, Inc., 1993.

[6] Horan, Stephen, "An Operational Concept for a Demand Assignment Multiple Access System for the Space Network".

[7] Motorola, "DSP56300 Family Manual, DSP56300FM/AD," 1995.

[8] Motorola, "DSP56303 Semiconductor Technical Data, DSP56303/D," 1996.

[9] Motorola, "DSP56303 User's Manual, DSP56303UM/AD," 1996.

[10] Motorola, "DSP56303EVM User's Manual," 1996.

[11] Motorola, "MC54/74HC04A Hex Inverter," 1995.

[12] National Semiconductor, "MM54HCT688/MM74HCT688 8-Bit Magnitude Comparator," 1995.

[13] Oppenheim, A. and Schafer, R. *Discrete-Time Signal Processing*, Englewood Cliffs, NJ, Prentice Hall, 1989.

[14] Quality Semiconductor, "Analysis of QuickSwitch Behavior in Four Operating Regions," Technical Note TN-07.

[15] Quality Semiconductor, "Bus Switches Provide 5V and 3V Logic Conversion With Zero Delay," Application Note AN-11A

[16] Quality Semiconductor, "QS3384, QS32384 High-Speed CMOS 10-Bit Bus Switches".

[17] Sanchez, M. and Horan, S. B., "Doppler Extraction for a Demand Assignment Multiple Access Service for NASA's Space Network," *NMSU Klipsch School of Electrical and Computer Engineering Technical Report Series*, August 1996.

[18] Scaife, Brad and De Leon, Phillip, "DAMA Data Collection/Test at the White Sands Complex," NMSU Klipsch School of Electrical and Computer Engineering, July 1998.

[19] Weste, N. and Eshraghian, K., Principles of CMOS VLSI Design, Reading, MA, Addison-Wesley, 1993.

Appendix A DSP Test Program Listing

AD_TEST.ASM

```
;*****************************************************************
; AD_TEST.ASM - Burr Brown ADS7819 to DSP56303EVM
; interface test driver
;
; This test code will initialize the DSP and A/D board, then perform
; a series of A/D data reads into a buffer. IRQB is used by the
; ADS7819 interface board to signal that a sample is ready
; to be read.


        nolist
; include 56302/3 standard equates from Motorola

        include     'ioequ.asm'
        include 'intequ.asm'

; include ADS7819 interface board equates
        include     '7819equ.asm'

; include AD_TEST.asm equates
        include 'test_equ.asm'

; include memory setup
        include 'test_mem.asm'
        list




        org   pli:
START
        movep #PLL_CTRL,x:M_PCTL      ;Initialize PLL

        movep #AAR0,x:M_AAR0    ;Initialize AA0 to ext SRAM
        movep #AAR2,x:M_AAR2    ;Initialize AA2 to A/D Board
        movep #BCR,x:M_BCR      ;Initialize Bus Control Register

        movep #IPRC,x:M_IPRC    ;Initialize Interrupt priority and
                                ;configuration
        move  #I_VEC,vba        ;Configure the vector base address


        move  #AD_BUF,r4        ;Point R4 to x: buffer block
        move  #BUF_SIZ-1,m4     ;Modulo addressing

        clr   a
        nop                         ;Pipeline stall
        move   a,y:FLAGS            ;clear all flags

        bclr   #3,x:M_PCRD          ;SCK ESSI1 pin GPIO
        bclr   #3,x:M_PDRD          ;Initialize output data to low
        bset   #3,x:M_PRRD          ;Make pin an output

        bclr   #0,x:M_PCRD          ;SC0 ESSI1 pin GPIO
        bclr   #0,x:M_PDRD          ;Initialize output data to low
        bset   #0,x:M_PRRD          ;make pin an output

        bset   #3,x:M_PDRD          ;Assert Port D pin 3 (SCK1) to
                                    ;enable logic analyzer capture
        andi   #$FC,MR              ;Enable all interrupts (levels 0-3)

AD_WAIT
        move   r4,a
        cmp    #BUF_SIZ+AD_BUF-1,a     ;R4 pointing at top of buffer?
        blt    AD_WAIT              ;No, loop again

        ori    #$03,MR              ;Disable all interrupts (levels 0-2)
        bclr   #3,x:M_PDRD          ;deassert Port D pin 3 to stop
                                    ;logic analyzer capture


        move   #($010000|(BUF_SIZ-1)),x0     ;Block size and file #
        move   #AD_BUF,r0                    ;Buffer base address
        move   #1,r1                         ;X memory
FILE1   debug                                ;Output command

        debug                                ;return to debugger

        jmp    *


; IQRB - A/D Interrupt Service Routine
; This fast interrupt will read data from the A/D interface board
; mapped into external Y: peripheral memory space, adn store the
; data sample into X: memory.
        org    pli:I_IRQB
;Read sample from A/D and store in mem
        movep y:BB7819_DR,y:(r4)+
        nop                               ;Waste 2nd fast int cycle


; IRESET - DSP Reset vector
        org    pli:I_RESET
        jmp    START

        end
```

TEST_EQU.ASM

```
; ***************************************************************
; TEST_EQU.ASM - equate definitions for the Burr Brown ADS7819
; interface board test software ad_test.asm.
;
; December 23, 1997
; Tim Baggett


; Define downsampling rate
N_DOWNSAMP      equ   2      ; 800/2 = 400 Ksps downsampled rate

BUF_SIZ         equ   $8000 ; Define sampled data block size

; Define Interrupt Vector Table p:$0000-$00ff
;I_VEC          equ   $0     ;Vector base address (defined in INTEQU)
          org   pli:I_VEC
          ds    $100         ;Reserve P space for Interrupt Table

; PLL Definitions
; The 56303 EVM has a 16.9344 MHz crystal on it. DSP clock will be
; defined by the PLL equation:
;
; DSP clock = ( 16.9 MHz * MF) / ( PDF * DF)

MF        equ   19     ;Multiply factor  (1-4096)
DF        equ   0      ;Divide Factor    (2^0 - 2^7)
PDF       equ   4      ;Predivide factor (1-16)


PLL_FREQ equ    (16.9344*MF)/(PDF*@POW(2,DF))
PLL_CTRL equ    ((MF-1)&M_MF)+(DF<<12&M_DF)+\
                ((PDF-1)<<20&M_PD)+(1<<M_PEN)


; Address Attribute settings
AAR0      equ     $010931    ;Unified X/Y SRAM 32Kword $010000-$017fff
AAR2      equ     $fffc21    ;Compare upper 12 address bits to $FFFxxx,
                             ;no byte packing, no address MUX,
                             ;Y enable, SRAM access

;Wait State Settings
DEFAULT_WS      equ   15          ;default are wait states (0-31)
SRAM_WS         equ   03          ;32KW SRAM (0-31)
FLASH_WS        equ   00          ;FLASH (0-31)
PERIPH_WS       equ   06          ;A/D Peripheral board (0-7)


AREA0     equ   SRAM_WS           ;AA0 wait states (0-31)
AREA1     equ   FLASH_WS          ;AA1 wait states (0-31)
AREA2     equ   PERIPH_WS         ;AA2 wait states (0-7)
```

```
AREA3     equ   DEFAULT_WS           ;AA3 wait states (0-7)


BBS       equ   0                    ;Bus State
BLH       equ   0                    ;Bus Lock Hold
BRH       equ   0                    ;Bus Request Hold


BCR       equ   (BBS<<21)+(BLH<<22)+(BRH<<23)+\
                (DEFAULT_WS<<16&M_BDFW)+(AREA3<<13&M_BA3W)+\
                (AREA2<<10&M_BA2W)+(AREA1<<5&M_BA1W)+\
                (AREA0&M_BA0W)

; CORE Interrupt Priority and Configuration
IBM       equ   1              ;IRQB trigger (0 level, 1 negative edge)
IBP       equ   2              ;IRQB priority level 0, 1, or 2

IBL       equ   (IBM<<2)+(IBP+1)&3

IPRC      equ   IBL<<M_IBL0&M_IBL


; FLAGS definitions. define bits in y:FLAGS as various flags used
BLOCK_ACQ       equ   0              ;Full downsampled data block filled
```

7819EQU.ASM

```
; ******************************************************************
; 7819EQU.ASM - equates for the Burr Brown ADS7819 to DSP56303EVM
; interface board
;
; December 18, 1997
; Tim Baggett
;

BB_ADR          equ   $0              ;DIP switch address on interface
                                      ;board for address decoder ($0-$3f)

BB7819_DR       equ   $FFFFC0+BB_ADR     ;ADS7819 Data Register
```

TEST_MEM.ASM

```
; TEST_MEM.ASM - Data memory configurations for AD_TEST.ASM
;
; December 26, 1997
; Tim Baggett


                org   yli:
FLAGS           ds    1           ;Flags


                org   xhe:
AD_BUF          dsm   BUF_SIZ     ; Define data buffer for A/D Samples

AD_BUF_TOP      equ   *
```

AD_TEST.MEM

```
base xli:$000000,yli:$000000,xhe:$010000,pli:$000000
```

MAKEFILE

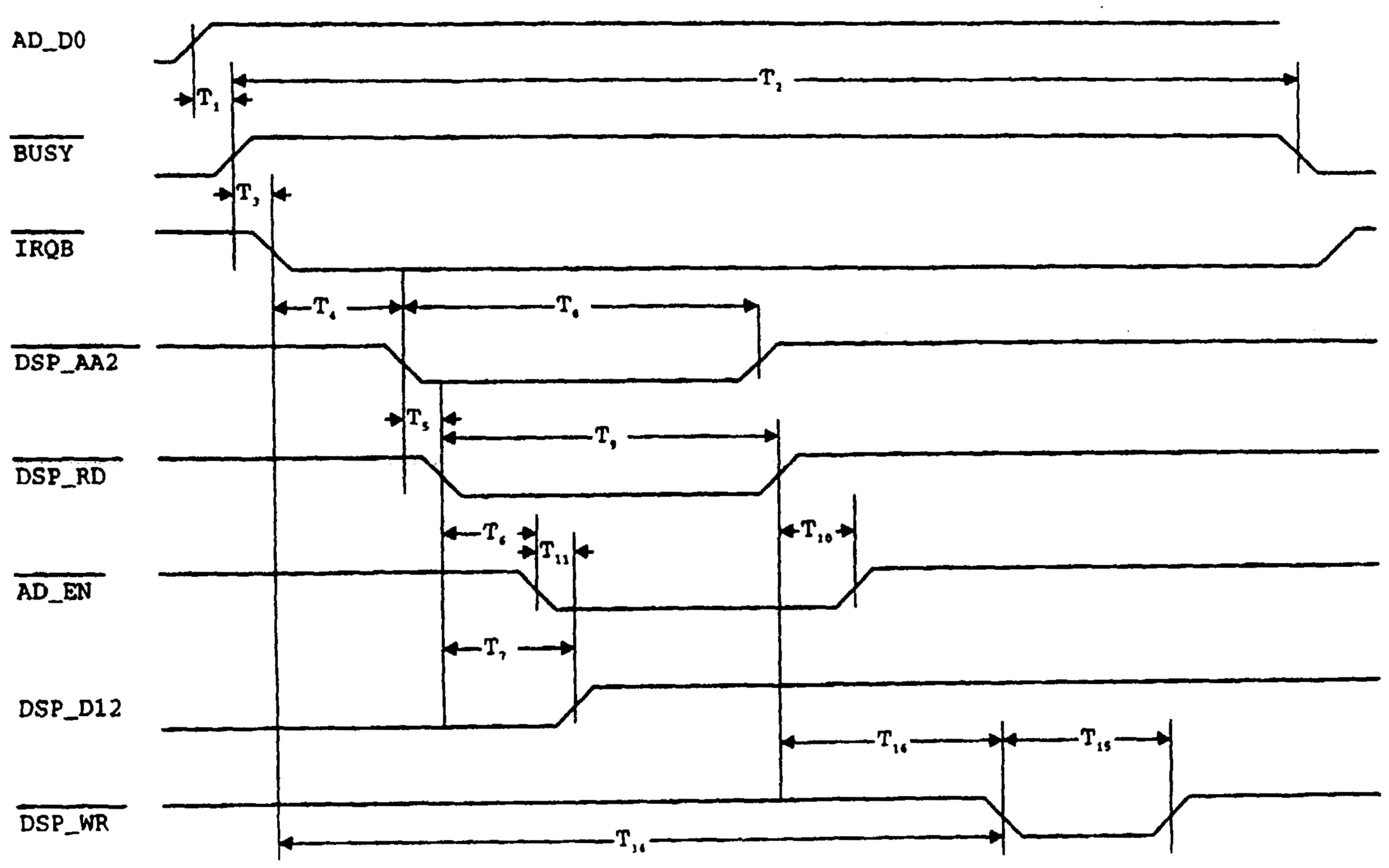
```
ad_test.cld: ad_test.cln ad_test.mem
   dsplnk      -b -m -rad_test.mem ad_test.cln

ad_test.cln: ad_test.asm 7819equ.asm test_equ.asm test_mem.asm
   asm56300 -b -l -g ad_test.asm
clean:
   rm *.o *.cld
```

This page intentionally left blank.

Appendix B Timing Diagrams

A/D Interface Board Signal Timing Diagram

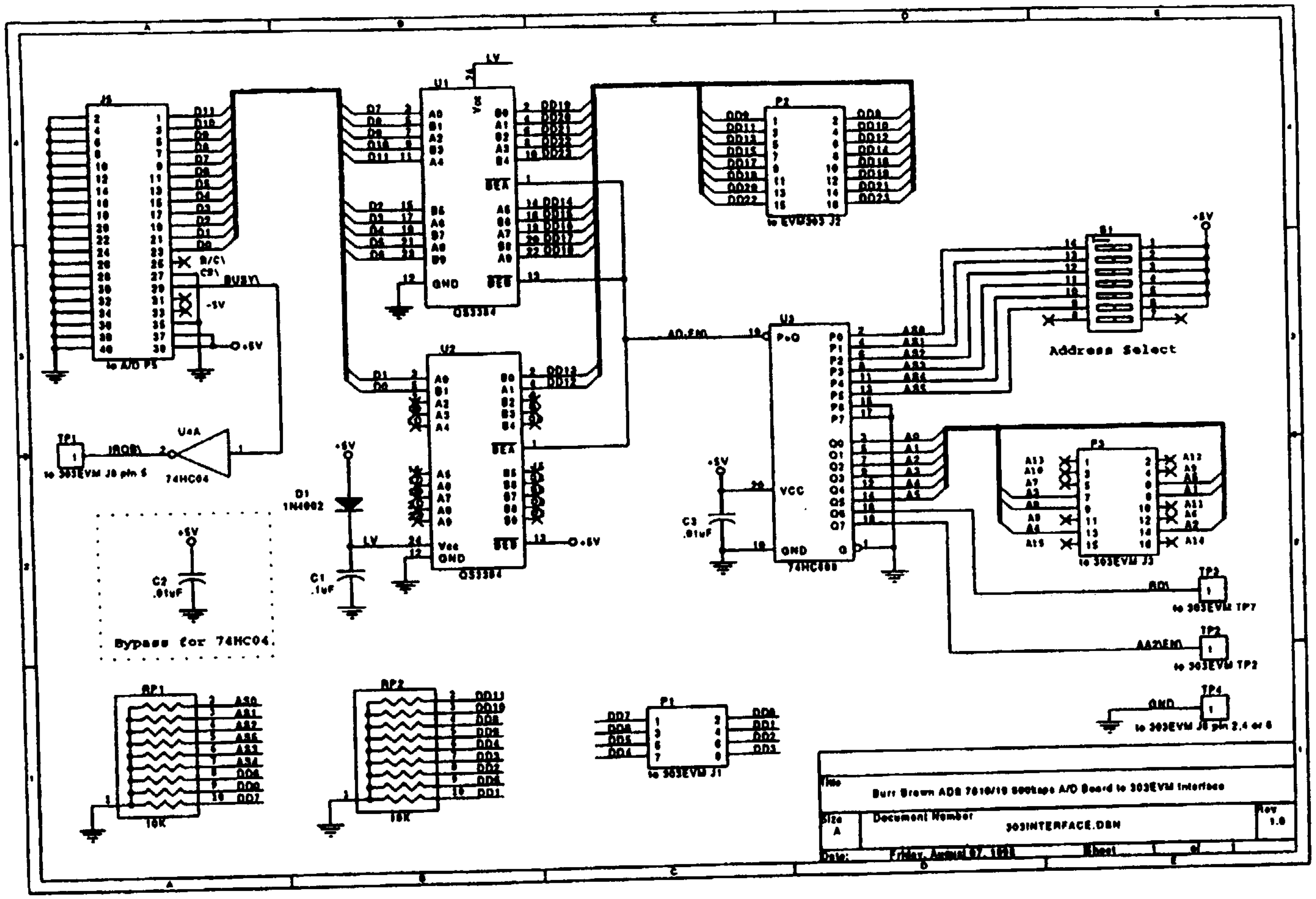


			6 Wait States Theory			6 Wait States Measured	
Description of timing measurement	**Timing #**	**min**	**typ**	**max**	**min**	**typ**	**max**
BUSY_ delay after A/D data valid (AD: t10)	t1	20	65	120		63	
BUSY_ deassertion width (AD: t12-t4)	t2	170	300		267	268.5	270
IRQB_ delay from BUSY_ deassertion (7404 Tphl)	t3		10	15	4.4	5.2	5.9
IRQB_ assertion to address valid (DSP: 17)	t4	88			97	116	135
RD_ assertion delay after address valid (DSP: 115)	t5	2.2			6.2	6.8	7.6
RD_ assertion to AD_EN_ assertion (74688 Tphl P or Q to output)	t6		23	44	20.7	25.7	27.3
RD_ assertion to input data valid (DSP: 105)	t7			68	21.8	29.5	35.5
DSP address valid width (DSP:100)	t8	95.5			99	99.2	99.5
RD_ assertion width (DSP: 116)	t9	74			77.1	77.9	78.2
Data hold time (74688 Tplh P or Q to output)	t10		16	30	15.4	16.3	16.9
AD_EN_ assertion to data valid (QS3384 Tpzh)	t11	1.5		6.5	1.1	3.8	8.2
A/D data not valid to BUSY_ deassertion (AD: t3,t9)	t12	-13	20	115			
BUSY_ assertion width (AD: t4)	t13		950	1080			
IRQB_ assertion to data sample WR_ assertion	t14	164.2			204.7	225.4	245.8
WR_ assertion width (DSP: 102) (3 wait states)	t15	176			36.2	36.8	37.1
Fast IRQ service time: t14+t15		340.2			240.9	262.2	282.9
RD_ deassertion to WR_ assertion	t16				24.4	24.7	25

Specified and Measured A/D Interface Board Signal Timings

- *All measurements are in nanoseconds*
- *DSP External SRAM (AA1) set to 3 wait states*
- *DSP A/D interface board (AA2) set to 6 wait states*
- *Calculated timings from technical specifications assume DSP clock at 80 MHz*
- *Measured timings with DSP clock operating at 80.4384 MHz*

Appendix C Schematic Diagram



Components List

Schematic ID	Part Number	Description
U1, U2	QS3384	QuickSwitch 10 bit bus switch (DIP)
U3	74HC688	CMOS 8-bit magnitude comparator (DIP)
U4	74HC04	CMOS Hex Inverter (DIP)
D1	1N4002	Diode with 0.7 volt forward bias
C1, C2, C3	0.1 uF	Ceramic 20 volt bypass capacitor
RP1, RP2	10K	9 Resister pack with common connection
S1		14 pin DIP switch (7 switches)
P1		4x2 pin male header, 0.1 inch
P2, P3		8x2 pin male header, 0.1 inch
J5		20x2 pin male header, 0.1 inch
TP1, TP2, TP3, TP4		1 pin male header